Lightweight Dependable Adaptation for Wireless Sensor Networks

Luís Marques
lmarques@lasige.di.fc.ul.pt
FC/UL

António Casimiro
casim@di.fc.ul.pt
FC/UL

Abstract

Achieving dependable and real-time operation in Wireless Sensor Networks (WSNs) is a hard and open problem. This can be an obstacle for many applications, namely in the automotive and medical domains, particularly if safety-critical control is envisaged. To overcome the communication uncertainties that are intrinsic to wireless and dynamic environments, a generic approach is to constantly adapt to environment conditions. This requires appropriate solutions to characterize such conditions.

This paper contributes with a lightweight solution for a dependable characterization of network QoS metrics, which is appropriate to support dependable adaptation in WSNs. The proposed solution offers probabilistic guarantees, building on non-parametric stochastic analysis to achieve fast and effective results. The paper also provides an evaluation of the solution.

Keywords

Wireless Sensor Networks; dependability; adaptation; non-parametric; lightweight; QoS;

I. Introduction

Wireless Sensor Networks (WSNs) are used to sense and collect the state of physical entities. Applications that use this kind of networks can therefore obtain a representation of the state of such entities, which they use in monitoring and control functions. Each application has its own requirements regarding how faithful such representation must be, compared with the true state of the sensed entity. The problem that arises in WSNs is the lack of guarantees relative to its timeliness. This means that applications have no guarantees regarding the degree of synchronization, or the consistency, between the sensor data they are using and the real state of the environment. Consequently, they cannot also guarantee that monitoring and/or control functions are performed in a timely way.

The lack of real-time guarantees in WSNs results from a variety of factors. These include, among others, uncertainties introduced by medium access control protocols, transmission interferences common to wireless communication, dynamic changes occurring in the network, such as node failures or their movement, and the lack of real-time behavior of the nodes themselves.

In practice, some of these factors, which we may call architectural factors, are under the designer control and thus their influence on uncertainty can be minimized or eliminated. For instance, nodes can be designed and built using traditional real-time operating system techniques, with real-time scheduling, and communication protocols can be made deterministic with respect to a set of assumptions on the communication medium. Still, there are a number of factors out of the designer's control, which we may call external factors or perturbations, that occur with uncertain patterns and lead to uncertainty, assumption violation and lack of real-time guarantees.

This work was partially supported by FCT through the Multiannual Funding and the CMU-Portugal Programs.

Note that these perturbations can be characterized as part of the fault model. For instance, omission faults, caused by electromagnetic interference or by physical obstacles, or temporal faults, caused by network contention and transmission back-off.

One possible way of dealing with these perturbations and improving timeliness characteristics of a WSN is by employing redundancy. A given amount of perturbations can be tolerated through some combination of multiple nodes at different locations, communication over different channels, multiple copies of the same messages, etc. This is particularly true if fault independence can be assumed, which is often the case.

Fortunately, WSNs do typically possess great amounts of redundancy, which can be employed to increase the guarantees of successful communication. On the other hand, one of the main concerns in this type of networks is to save energy, and thus radio transmissions should be avoided as much as possible, as they are one of the main sources of energy consumption. That being, the operational objective of sensor networks should be that of satisfying application requirements through the use of the strictly necessary redundant resources. In other words, simply adding a fixed amount of redundant resources is not the best solution since it just pushes the bounds further, allowing a larger tolerance of perturbations, but not necessarily the adequate or sufficient one. Achieving some level of dependability and efficiency requires some effective way of dealing with the mentioned perturbations, whose distribution is a priori unknown and for which might not even be practical to define an arbitrary limit.

From a theoretical standpoint, not having a guaranteed limit on the amount of perturbations that can occur on WSNs makes it impossible to offer hard real-time guarantees. From a practical standpoint, even if the amount of perturbations never reaches the maximum amount that a sensor network can tolerate, it is not desirable to constantly use the maximum amount of redundant resources available. One alternative to offering strict hard real-time guarantees is to offer, instead, probabilistic guarantees. Therefore we consider that perturbations occur in a probabilistic manner. By analyzing the statistical behavior of such perturbations we can derive the necessary amount of redundancy that is necessary to satisfy application requirements, with a given probability. Since this behavior can change throughout time, as environment conditions change so must the system adapt, so that application requirements are continuously satisfied. In this paper we consider how to implement such statistical analysis and adaptation in a manner that is appropriate for WSNs.

We propose an approach for monitoring and raising awareness of communication latency that is based on non-parametric stochastic analysis. From a complexity perspective, the solution is extremely simple and thus appropriate for resource-constrained systems, such as WSNs. Despite its simplicity, it builds upon established theory on stochastic systems and allows probabilistic guarantees to be asserted to relevant temporal bounds. To conclude about the potential merits and the effectiveness of the proposed approach, we compared it with another solution, called Adaptare [1], which is based on complex and expensive stochastic analysis mechanisms. To ensure a fair comparison, we used the same data traces that were previously used for validating Adaptare, and we observed that our approach is very effective in general, although it exhibits some comparative limitations when the required probabilistic guarantees are very high. We believe that this trade-off may be acceptable in the context of WSNs, given that it still provides valuable monitoring data at an extremely low-cost.

The paper is thus organized. In the following section we provide some context and introduce various concepts. Then, Section III presents the idea of lightweight adaptation based on non-parametric analysis. In Section IV we evaluate and benchmark several aspects of the solution and in Section V we present related work. Section VI concludes the paper.

II. Problem Context

Dealing with predictability or real-time requirements in WSNs is a difficult and open problem, which may be addressed from several perspectives. While the provision of strict real-time requires satisfying very stringent assumptions, focusing on the provision of some Quality of Service (QoS) is a reasonable alternative to address the problem. In fact, several authors have explored the issue of providing QoS guarantees in WSNs, which we review in more detail in Section V. Our work is also developed in this context.

Proposed solutions explore topological aspects, such as end-to-end path discovery, resource reservation along discovered paths and path recovery from topological changes, build on proper scheduling of real-time traffic (e.g. decentralized EDF scheduling), and define efficient network protocols to support the necessary QoS features. Such solutions deal mainly with problems that are architectural and internal to WSNs, namely how to match the low-power and dynamic nature of WSNs with the strict requirements of various types of real-time communication. Another body of work has explored how to make radio communication efficient and resilient. Employed techniques include detecting interferences [2] and dynamically changing channels to avoid them [3], [4].

But no matter the techniques and approaches that may be used to endow WSN-based systems with better predictability and greater ability of dealing with user QoS requirements, we believe that the likely occurrence of perturbations still makes it inappropriate to specify fixed upper bounds on system variables, such as the maximum number of omission faults, latencies, query load, and so on. Because of that, monitoring and adaptation appear to be relevant techniques to deal with this uncertainty, contributing to achieve systems that perform more closely to the allowed environment conditions and thus leading to improved QoS.

Quite clearly, one fundamental issue in this context is how QoS is defined. Different metrics (or estimators of metrics) can be considered, and they should be in some way related to application requirements. Possible metrics and estimators may include: rate of arrival of updates, number of duplicate packets received, jitter in the arrival of updates, packet loss rate, Packet Error Rate (PER), Received Signal Strength Indicator (RSSI), and single hop or end-to-end latency.

From the application perspective, QoS requirements tend to be less functional. For instance, for the correctness of monitoring and control applications what is essential is to ensure that real-time sensor data is accurate, close (within some error interval) to the real value of the monitored or controlled entity. We refer to this as a requirement for *perception quality*, that is, how accurately the application perceives the reality. Given the uncertainties affecting WSNs, and being inappropriate to assume fixed upper bounds for network latencies, the notion of perception quality encompasses both the acceptable error for sensor data and the probability that this error bound will be secured at run time. In summary, high perception quality means ensuring a very small perception error with a very high probability. We also say that the assumed error bound is secured with a certain *coverage*, i.e. the probably of the observed value being within the required error margin.

However, as mentioned above, in practice it is usually necessary to translate higher-level, possibly non-functional, application requirements to observable metrics. In this paper we consider that the propagation latency is the relevant QoS metric. Without loss of generality, we consider as a simplifying assumption that there is an inverse proportional relation between end-to-end latency and perception quality.

III. Lightweight Dependable Adaptation

We consider that the environment behaves as a stochastic process. That means that each relevant QoS property of the system is defined (at a given instant) by a random variable. Such random variable describes the values that the property can take, and with which probabilities — that is, its probability distribution. As time progresses and the environment changes, new random values will take the place of old ones, in ways characterized by the stochastic process.

For adaptation to occur it is necessary to characterize environment conditions. To deduce at a given time what is the state of the environment it is necessary to first sample its behavior. Using a sample we can then make inferences regarding the state of the system, and in particular estimate the probability distributions of the random variables.

In the same way that there are an infinity of numbers, there are also an infinity of possible probability distributions. In practice, some numbers are particularly common or important and thus become well-known, such as the numbers 1, 2, π or $\sqrt{2}$. Likewise, there are various well-known families of probability distributions, such as the Normal, Exponential and Poisson distributions. Despite their name, each well-known "distribution" does not specify a particular probability distribution function. Instead, they have parameters which control properties of the distribution, such as their average value or dispersion, giving rise to an infinity of fully specified probability distribution functions.

Traditionally, the process of statistical inference is done through parametric methods. That is, methods which assume that the sample values are the result of a random variable whose distribution function belongs to a well-known family, but for which the parameters must be estimated. This assumption, when valid, brings several benefits. It allows estimators to have more *statistical power*, producing estimates that are more accurate or that otherwise could not even be determined. The simplicity and benefits of parametric methods made them widely applied, and they are often used even when no distribution perfectly matches the studied phenomenon. However, when the assumptions made by parametric statistics do not hold the results can be extremely misleading. Because parametric methods strongly depend on their assumptions they are not statistically *robust*.

Prior work applying statistical adaptation [1] has tried to guarantee the parametric assumptions by employing runtime statistical diagnostics. That involves checking the *goodness of fit* of the observations of a random variable against the possible distributions. If the fit is good enough (i.e., very unlikely to be due to chance) then parametric methods can be used safely. Unfortunately, the disadvantages of such strategy are particularly onerous for WSNs. Most importantly, the necessary statistical tests are computationally expensive, making them unfeasible for limited devices such as sensor nodes.

As an alternative, we propose to achieve lightweight dependable adaptation through the use of non-parametric statistics — that is, methods which are distribution-free. Non-parametric methods are robust, since they apply to all probability distributions, making them fit for the wide variety of WSN scenarios and protocols. They are also lightweight, requiring only very simple computations, thus conserving energy. And, despite their simplicity, they can also be surprisingly effective.

A. Non-parametric statistical analysis

The fundamentals of how non-parametric statistics are used in the proposed solution, and how they differ from the parametric methods of previous work, can be explained by comparing the two histograms in Figure 1.

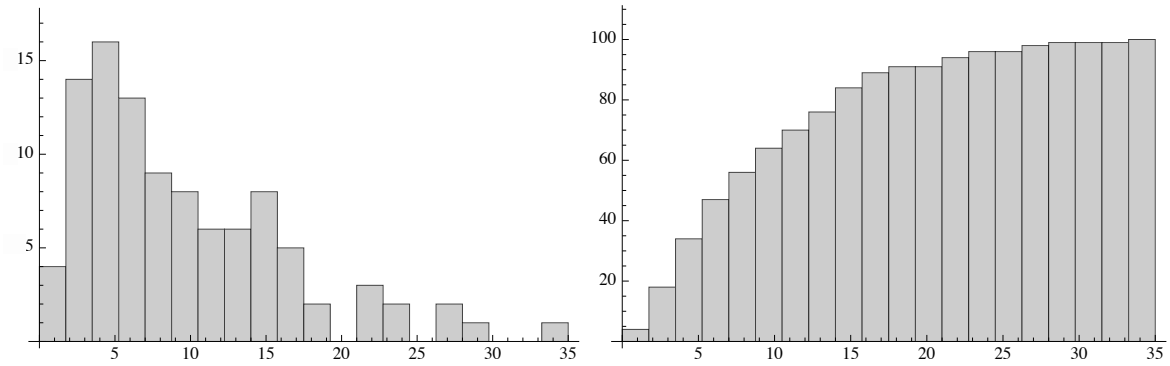

Figure 1: Two histograms for one same sample: normal and cumulative

The top histogram of Figure 1 shows twenty bins, each of which counts the frequency of values occurring in a particular interval. By glancing at such an histogram we can informally check if there is a good fit between the sample values and well-known distributions. In this case we do not immediately recognize some of the most well-known distributions: the distribution is asymmetric, so it is not a Normal distribution; after the third bin the histogram resembles an Exponential distribution, but the first two bins do not match; it is also not quite the shape of a Poisson distribution with a low λ value.

If we included a near-infinite amount of distributions against which to test the goodness of fit of our sample values then, eventually, we would find a good match. But such method would not achieve good results: the performance of testing huge amounts of distributions would not be sustainable on a sensor node. We would also match many possible distributions, so we would not know which to use to better predict yet unobserved values and their probabilities.

An alternative to matching the sample values to a theoretical model, and then using the properties of the theoretical model to drive adaptation, is to use only the statistical properties of collected sample itself, free of distribution assumptions. The bottom histogram of Figure 1 illustrates that approach, by virtue of being a *cumulative* histogram.

In a cumulative histogram the bins count the occurrences (or relative frequency) of values that fall in the range of those bins *or the preceding ones*. It thus becomes easier to see what percentage of the sample values is equal or smaller than some other value. For instance, we see that about 80% of the sample values are equal or smaller than 15. If this were a sample of latencies in the system then we could use this non-parametric analysis as statistical evidence to drive the adaptation. For instance, we could make sensor nodes wait for some event only up to 15 time units, and that way know that timing failures would occur only for approximately 20% of the events. That is, we would be using the empirical distribution to guide the process of adaptation.

Directly using the empirical distribution to make statistical inferences about the sampled system disregards the problem of sampling error. For instance, by using the empirical distribution present in Figure 1 we might be tempted to conclude that a time bound of 35 time units would be enough to receive 100% of events, and that way avoid timing failures. Of course, this ignores the possibility that there might be a small percentage of events which have higher latencies but, by chance, were not captured in the sample.

One possible solution then is to increase the sample size. As the number of sample values approaches infinity the statistics of the sample converge to the true values of the population.

Alas, it is not practical to use nearly infinite sample sizes. For one, it would be computationally expensive, particularly in sensor nodes. Also, as the environment changes the older values would no longer reflect its state.

The solution, therefore, must entail taking into account the sampling error and estimating how much of the population really is equal or less than a given sample value. In fact, this is not done for the sample value per se but, instead, for its *ordinal ranking* in the sample. So, in the example given before, 35 time units was estimated to cover 100% of the population not because the value was 35 but because that was the n^{th} order statistic of a sample with n values (i.e. the maximum)[1]. What we want, then, is to make such inferences but taking into account the sampling error, and for a generic order statistic. For that we can use the Beta distribution.

The k^{th} order statistic of the Uniform distribution can be described by a Beta(α, β) distribution with parameters $\alpha = k$ and $\beta = n + 1 - k$. The mean of the Beta distribution is given by:

$$\text{mean} = \frac{\alpha}{\alpha + \beta} \tag{1}$$

Therefore, the mean for the k^{th} order statistic is given by:

$$\text{mean} = \frac{k}{n + 1} \tag{2}$$

Consider then a sample for a Uniform(0, 1) distribution, with $n = 20$. In that case the order statistic $k_{(20)}$ would take, on average, the value of $\frac{20}{21}$, or approximately 0.95. If we used the order statistic as an upper bound we could then infer to be covering, on average, 95% of the population, instead of the 100% we would expect while not taking into account the sampling error.

We could take such a direct conclusion because for a Uniform(0, 1) distribution the rank of a quantile is equal to the value of the quantile itself. For instance, the quantile $q_{0.25}$ represents the value x such that the probability of a random variable being less than or equal to x is 25%. For a Uniform(0, 1) distribution the value x is also 0.25, the same as the rank.

For other distributions the values are not uniformly apportioned between 0 and 1. Nevertheless, the quantiles *are*, by definition, uniformly distributed between 0 and 1. As such, it is still correct to use a Beta distribution to infer what is the average rank of the quantile to which an order statistic corresponds. The next section explains how to generically apply this to a monitoring solution.

B. Monitoring Method

The overall process of adaptation is done by sampling the environment, inferring the state of the system from the sample values and adapting to that environment. The adaptation itself can start as soon as enough values are collected. The necessary amount of sample values depends on the desired (or target) average coverage.

On the one hand, it depends on the *minimum* desired coverage. The n^{th} order statistic has an average value of $n/(n + 1)$. As such, with 1 sample value it is possible to start the adaptation for minimum coverages of up to $1/(1 + 1) = 50\%$, with 2 values for minimums of up to $2/(2 + 1) = 66.7\%$, with 3 values for up to 75%, and so on.

[1]The i^{th} order statistic of S is the i^{th} smallest element of S. Such an element is said to have *rank* i.

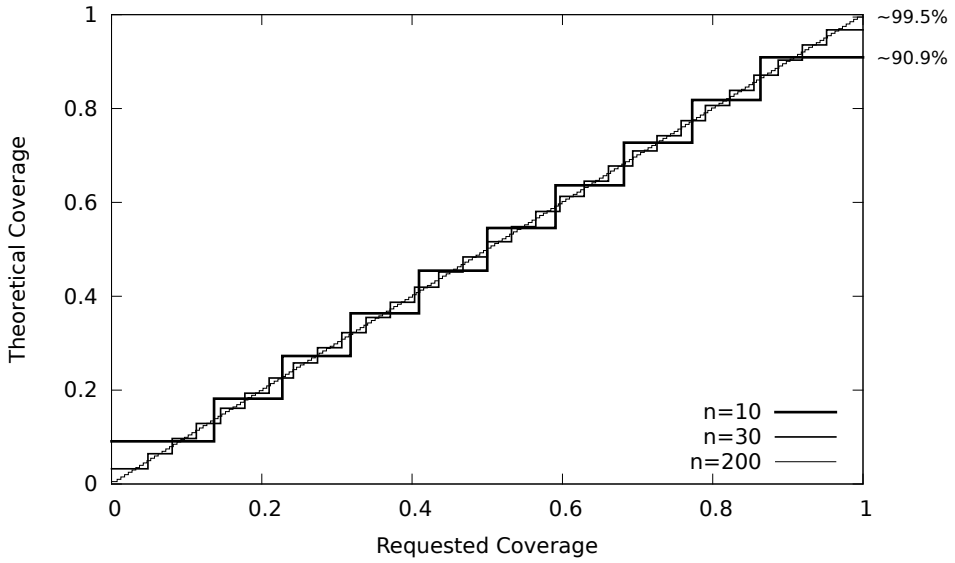

Figure 2: Non-parametric average coverages for different sample sizes

On the other hand, the number of required sample values also depends on how *accurately* the coverage must be matched. Figure 2 illustrates (with lines of different widths) the impact of choosing different sample sizes on the average coverage obtained by applications.

Ideally, the process of adaptation would result in applications obtaining a coverage equal to the one asked by the application. What we see is that with a sample of only 10 values there are large gaps between various coverages the application can ask for and the coverage that is obtained. Also, even by choosing the largest value from the sample the application will only obtain (on average) a coverage of 90.9%, which is not very high. On the other hand, for a sample with 30 values the gaps between the requested coverages and the average ones become more reasonable. Finally, for a large sample of 200 values we observe that the adaptation process starts to approach an optimal diagonal line, and can reach a high coverage of about 99.5%.

Having a sufficient number of sample values, we then order the sample and chose the value whose rank best matches a target average coverage. If we take equation 2, substitute the mean by a target coverage C and solve for the rank k we get:

$$k = C * (n + 1) \tag{3}$$

This should give us the rank of the sample value which matches the target coverage C. Unfortunately, the equation does not consider two problems. One is that the derived rank might be non-integral, if no rank exactly matches the requested average coverage. Another is that the computed rank may be out of bounds, if the sample size does not achieve a low enough or high enough average coverage.

Algorithm 1 takes the base calculation of equation 3, performs the necessary adjustments to deal with those two problems, and returns the sample value whose rank best matches the requested coverage. It is also adjusted for a zero-based indexing of the sample array, for additional clarity of implementation.

While coverages of exactly 0% and 100% would never be valid for samples of finite size, Algorithm 1 accepts target coverages with values of 0 and 1. Since floating point numbers have

Algorithm 1 Bound estimation algorithm

Input: ordered sample array $sample[n]$
Input: desired average coverage C (floating point)
Output: time bound (with average coverage C)
1: assert ($C \geq 0.0$ **and** $C \leq 1.0$)
2: $index \leftarrow \lfloor (C * (n+1)) - 0.5 \rfloor$
3: **if** $index < 0$ **then**
4: $\quad index \leftarrow 0$
5: **else if** $index \geq n$ **then**
6: $\quad index \leftarrow n - 1$
7: **end if**
8: **return** $sample[index]$

finite dynamic range, similar values may happen to be rounded to those limit values. With the addition of the guard conditions on lines 3–7 such cases are gracefully handled anyway.

After the minimum sample size is attained we can continue to append new values to the sample. The maximum sample size should be chosen according to the limitations of the hardware and the dynamics of the environment. For rapidly changing environments smaller sample sizes must be chosen, to ensure that the adaptation process does not reflect stale views of that environment. Section IV-C evaluates the empirical impact of different sample sizes. Once the sample is filled to its maximum the oldest values should be discarded before adding new values. This results in a sliding window.

IV. Evaluation

A. Complexity Analysis

Non-parametric order statistics entail selecting a k^{th} smallest value from an ordered sample. Our sample is a constantly changing (unordered) sliding window, updated from the latencies of received packets, up to a size n. This sliding window can be implemented as circular buffer, with an array for the sample values, a pointer to the beginning of the window and a fill counter. This way, adding new values to the sample takes $O(1)$ time, with a very low constant.

An obvious way to select the k^{th} smallest values from the unordered sample is to create a copy of the sample, sort it, and select the k^{th} value. Such solution has a worst-case time complexity of $O(n \log n)$ and a space complexity of $O(n)$. For small sample sizes this cost can be reasonable, even on a limited device such as a sensor node. A sorting function can generally be reused from the sensor node's software library, not incurring an extra code space penalty.

For larger sample sizes a better alternative is to use the selection algorithm first described in [5] and more clearly explained in [6], which has a $O(n)$ worst-case time complexity and $O(1)$ space complexity. This algorithm is, on average, less efficient than Hoare's Selection Algorithm, but is safer for real-time sensors, since the later algorithm has a nonlinear worst-case time complexity, making it less predictable.

If sublinear time complexity is required then a red-black tree can be used, at the expense of additional memory and an increase in insertion time. Updating the tree-based ordered sample takes time with $O(\log n)$ complexity, instead of $O(1)$, but the order statistic can be found in $O(\log n)$ time.

B. Performance Magnitude

Today's sensor nodes are equipped with very rudimentary CPUs. While that is expected to eventually change, for now that is a reality that must be dealt with.

Besides being slow, the CPUs of sensor nodes also typically lack a floating point unit. While it is possible to add floating point software emulation libraries, such libraries would make computations even slower. They would also take up a lot of code space, which is very limited on sensor nodes, generally on the order of a few kilobytes. Since the general bound selection algorithm provided in Algorithm 1 relies on floating point, for practical implementations it must be optimized.

One possible alternative is to use fixed-point arithmetic. For instance, instead of specifying a coverage of 85% as the floating point number 0.85 it can instead be specified by the integer 85. Such alternative algorithm can be implemented using an integer division, plus a few basic operations.

Still, even an integer division is a complex operation. In fact, the CPU of the popular MICA2 mote (an Atmel AVR ATmega128) does not even implement a native integer divide instruction. One possible solution is to use a software division routine. For that CPU, a 16 / 16 bit division, with 16 + 16 bit signed result can be implemented in 39 code words, taking 255 execution cycles to compute (16 bits allows for samples with more than 256 values).

Another possibility, which avoids division instructions, is to receive the input coverage as an integer in the range [0, 127], instead of [0, 100%]. Algorithm 2 exemplifies how such an optimized bound estimation algorithm can be implemented.

Algorithm 2 Fixed-point bound estimation algorithm
(coverage range [0, 1] mapped to {0, 1, ..., 127})

Input: ordered sample array $sample[n]$
Input: desired average coverage C (integer)
Output: time bound (with average coverage C)

```
1: assert (C ≥ 0 and C ≤ 127)
2: index ← ((C * (n + 1)) − 128)
3: index ← index/128
4: if index < 0 then
5:     index ← 0
6: else if index ≥ n then
7:     index ← n − 1
8: end if
9: output sample[index]
```

While Algorithm 2 shows a division by 128 at line 3, that division can be implemented as a right shift instruction, and thus be computed very quickly.

With the presented simplifications, the algorithm requires on the order of less than 100 simple instructions. To this we must add the additional cost of sorting the sample, or of using a linear time selection algorithm. Assuming the worst-case cost of $5.4305*n$ comparisons for the selection algorithm reported in [5], also assuming about 5 cycles per comparison, and a sample size of 30 values, we would be adding 815 cycles. That is still well under 1000 cycles total. We can

therefore estimate a performance magnitude of more than 8,000 bound estimations per second, for a simple 8 MHz CPU.

As such, we conclude that the proposed adaptation algorithm can be made lightweight enough for Wireless Sensor Networks. It is not a bottleneck and should not have a significant impact on energy consumption.

C. Adaptation Benchmark

The statistical properties in which we rely for monitoring the environment and driving the adaptation process are theoretically guaranteed. In practice, they only hold if the assumptions in which they are based can be relied upon. Particularly, we assume that the behavior of the environment is stochastic, with limited dynamics.

In this section we benchmark an adaptation process, which uses the proposed bound estimation algorithm, to determine the empirical effectiveness of the non-parametric method. The benchmark verifies if, given a stream of real network latencies, the estimated bounds are able to maintain the desired average coverage. We selected the following collection of network latency traces for the benchmark:

- **Inmotion**: FTP file transfers between cars traveling at various speeds and an 802.11b access point;
- **Umass**: Wireless traces from University of Puerto Rico, using laptops over various distances;
- **Dartmouth**: Wireless traces from Dartmouth College;
- **LBNL/Datcat**: Traffic of an enterprise network from Lawrence Berkeley National Laboratory (LBNL);
- **RON**: Latency samples from the RON (Resilient Overlay Network) testbed;

These traces were previously used to benchmark Adaptare, an adaptation framework based on parametric methods, and are referenced in [1]. Using these specific traces has some advantages. Because we are reusing the same traces that were used to benchmark Adaptare, it allows us to accurately compare our results, with regards to the obtained coverage. The traces also exhibit a great variety of behaviors. Since the field of WSNs is still evolving rapidly (with no de facto standard for MAC and network protocols), and sensor networks can be applied in very different scenarios, it is important to test if the proposed method is effective for a variety of latency patterns.

In Figure 3 we plotted the chosen traces, showing the evolution of latencies throughout time. No scale was included for simplicity, but the figure allows us to visually confirm that the traces *do* exhibit a wide variety of patterns, and therefore are suitable to validate the proposed method.

There is necessarily a trade-off (all things being the same) between higher bounds, which can guarantee a higher coverage but will result in lower performance, and lower bounds, which provide lower coverages but also allow higher performances. As such, this benchmark tries to evaluate the capability of the proposed algorithm to estimate bounds which best match a requested coverage, with its implied trade-off.

One trial of the benchmark was performed as follows. We evaluated both the proposed non-parametric method and the Adaptare framework, for each trace. We selected the most favorable sample size to Adaptare, 30 values [1], to assure a meaningful comparison, despite that sample size limiting the maximum coverage of the non-parametric method to about 96.77%. For each method, we examined a range of target coverages, from 1% to 99%, in 1% increments. For each trace, the first 30 values were used to fill the sample, leaving the other trace values to test the bound estimation. For each of the remaining trace values we repeated these steps: 1)

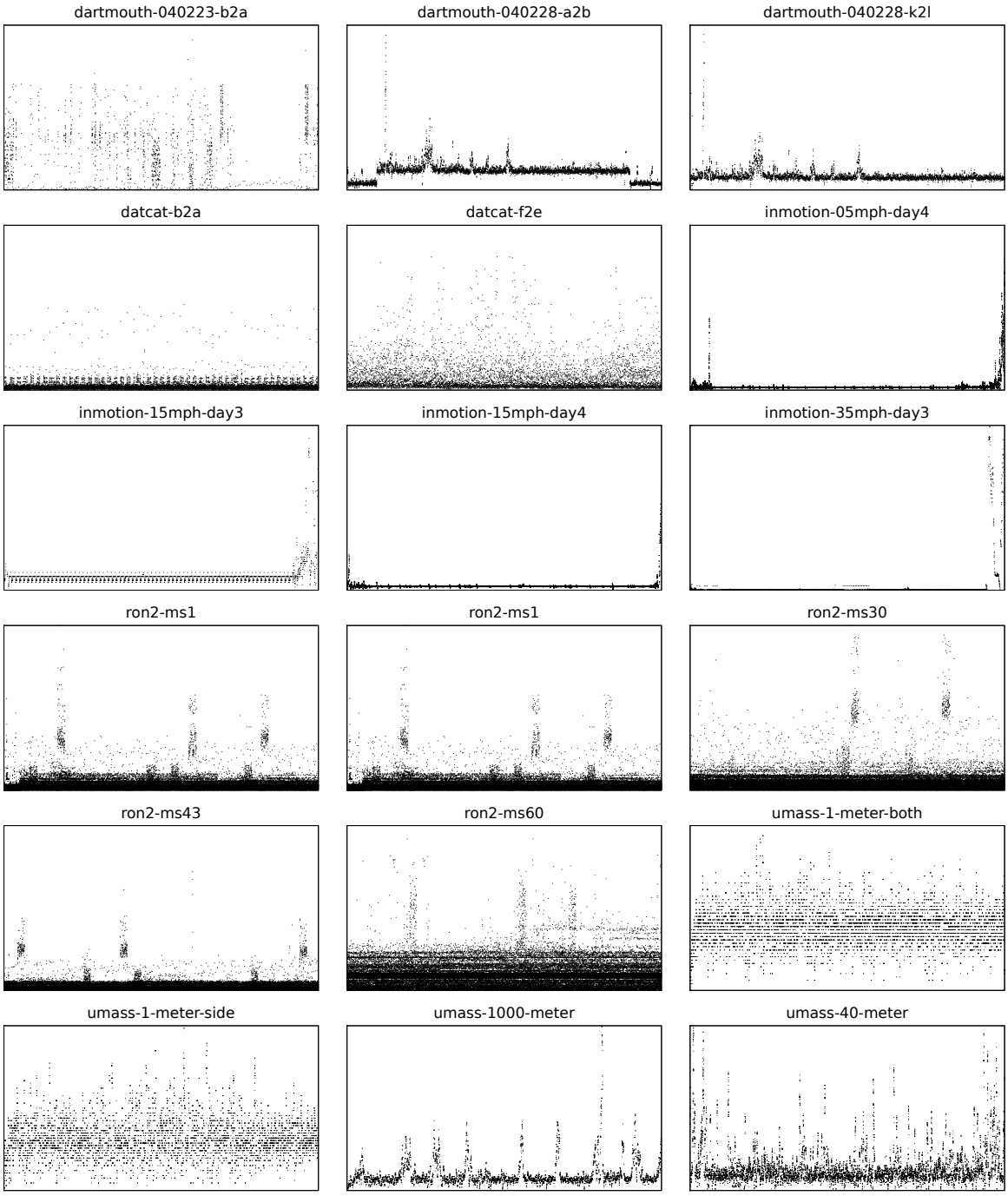

Figure 3: Plots of recorded latencies, exhibiting different dynamics

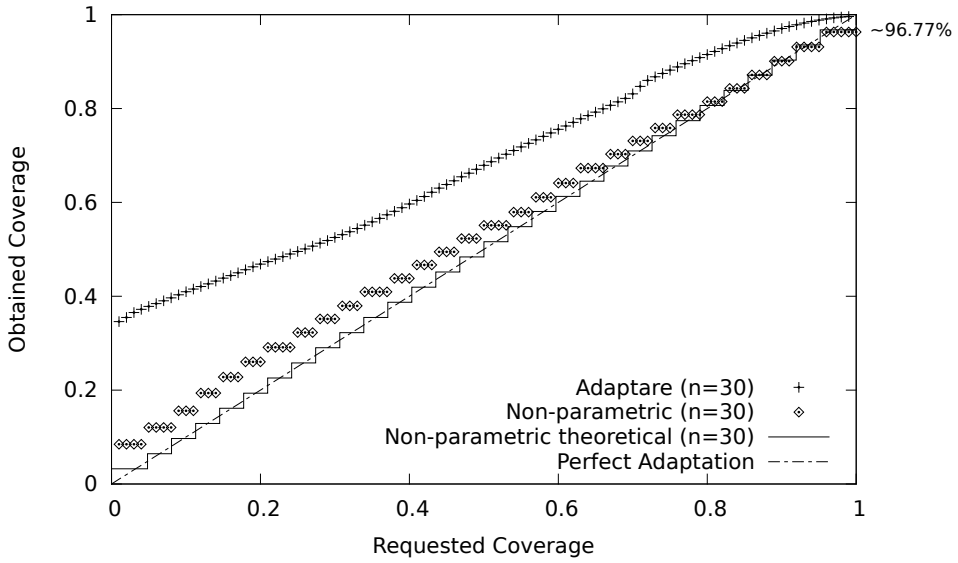

Figure 4: Empirical average coverages ($n = 30$)

estimated latency bounds using both the non-parametric method and Adaptare; 2) increased timeout counters for each method, if the trace value exceeded the estimated bound; 3) removed the oldest value from the sample; 4) added the new trace value to the sample. At the end of the process we computed the obtained average coverage for each method, using the formula:

$$\text{coverage} = 1 - (\text{timeouts}/\text{tested\_samples}). \tag{4}$$

Figure 4 plots an average of the coverages that were obtained for all the tested traces. Looking at this figure we can take the following conclusions:

- Both the proposed non-parametric method and the Adaptare framework are able to adapt to different target coverages, increasing the empirical coverage for higher target coverages;
- Despite its simplicity, the non-parametric method comes much closer to matching the target coverage (the "perfect adaptation" line), especially for higher coverages;
- Due to the small sample size the non-parametric method exhibits a significant staircase effect. These is the result of choosing the same order statistic for different target coverages, and is to be expected;
- Also due to the chosen sample size, the non-parametric method is unable to achieve the highest target coverages. Contrarily, the Adaptare framework can deduce statistical properties of the environment not directly present in the sample, to successfully estimate bounds for the highest coverages.

Figure 4 summarizes the results for the tested traces, by presenting an average of those results. It is legitimate to question what variation there is in the individual results. Figure 5 and Figure 6 present the best and the worst individual results, respectively.

We see in Figure 5 that the non-parametric method achieves nearly perfect adaptation, while the Adaptare framework retains a behavior similar to the average. In Figure 6 we observe that both for the non-parametric method as for the Adaptare framework there is some disruption in

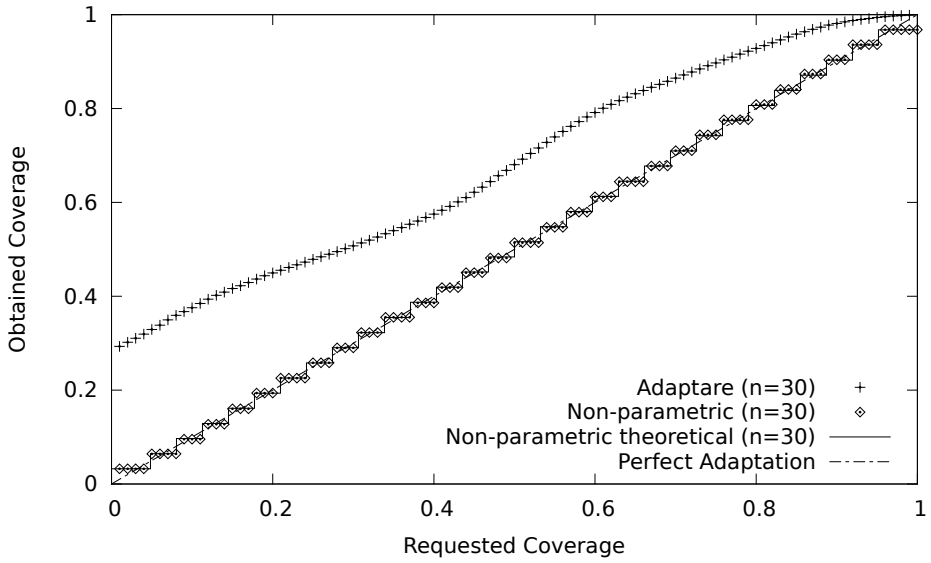

Figure 5: Empirical coverages: 'ron2-ms60-30'

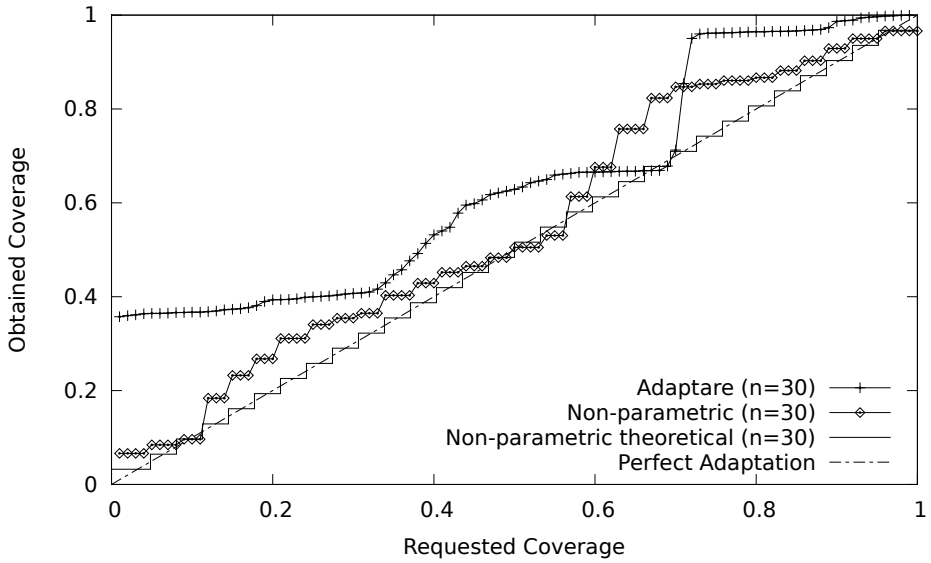

Figure 6: Empirical coverages: 'inmotion-15mph-day3-30'

the adaptation, compared to the averages. Still, the non-parametric method is on average closer to the target coverage.

Observing Figure 3 we note that the latency plots for the RON traces do exhibit a seemingly stochastic behavior, while the traces for Inmotion have repeating patterns (e.g. inmotion-15mph-day3) and other forms of low randomness. Also, while the RON traces reflect the prolonged interaction of many different network nodes, the Inmotion traces capture brief file transfers between a single mobile node and a base station, distant from other active network nodes. Therefore, we believe that the differences in the obtained results reflect how well the different scenarios meet our assumptions.

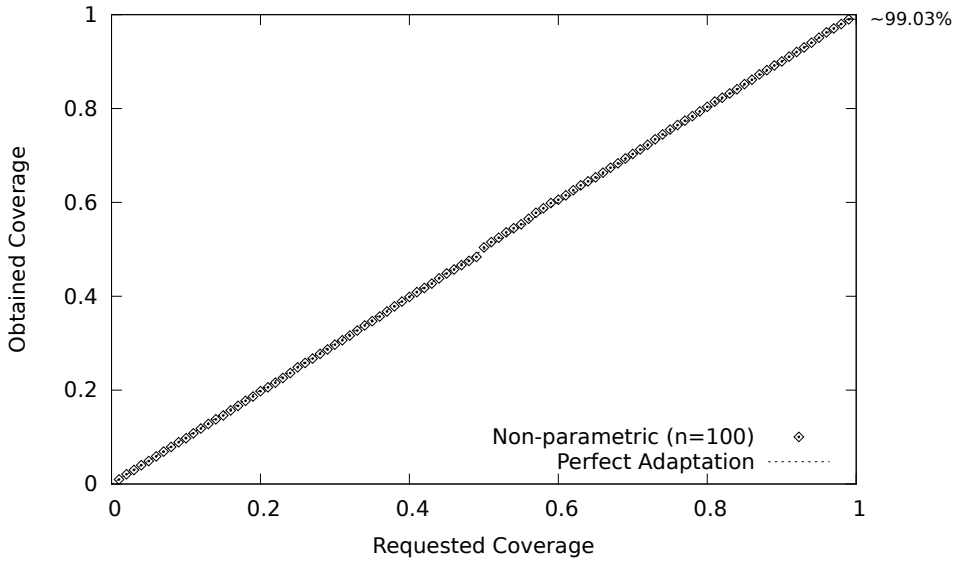

Figure 7: Empirical average coverages ($n = 100$)

Since we aim to apply the presented adaptation methods to WSNs, which typically have many nodes and operate in open environments, we believe that these results validate the adaptation capabilities of the non-parametric method. In scenarios which closely match our assumptions the results show a nearly perfect adaptation, while in adverse scenarios the method still achieves coverages generally close to the target. For the favorable scenarios the only significant limitations that were found were due to the limited sample size. For that reason, another trial of the benchmark evaluated a larger sample size.

Figure 7 presents the coverages of the non-parametric method, averaged from all traces, using a large sample: 100 values. We observe that, as expected, a larger sample size raised the maximum achievable coverage, to more than 99%. The larger sample size also removed the staircase effect and resulted in empirical coverages much closer to the target. For instance, the maximum coverage obtained was 99.07%, close to the theoretical average of 99.01%. These results also validate our assumption of limited environment dynamics, since with a larger sample (i.e. with one which includes older values) we still achieved a very good adaptation — in fact, better than with a smaller sample.

V. Related Work

There are two important lines of work related to this paper. One concerns protocols and other architectural mechanisms which aim to achieve real-time or QoS-driven behavior in WSNs. Another one deals with reliable wireless communication, including the physical mediums, radio propagation, link quality, interferences and similar issues. In this section we review related work on these subjects.

A. QoS and Real-Time

Two surveys of WSNs which summarize various QoS and real-time related issues can be found in [7] and [8]. These issues include the network architecture, platform and sensor hardware,

collaboration and coordination protocols and middleware, as well as MAC, network and transport layer protocols.

The work presented in [9] focuses on issues related to traffic QoS, and on various routing and MAC protocols that were proposed to address QoS requirements.

For exploiting the redundancy of WSNs, multi-path protocols have been proposed, for instance in [10]. Also relevant for QoS provisioning, the *Multi-Path and Multi-Speed Routing Protocol* (MMSPEED) is described in [11]. It tries to achieve QoS differentiation in two independent domains: *timeliness* and *reliability*. To achieve the necessary reliability, MMSPEED also attempts to exploit the redundancy of dense sensor networks and uses multipath forwarding. Differently from our approach, MMSPEED uses rudimentary mathematical calculations that are not supported by a statistical method. The end-to-end communication success probability is extrapolated at each node by keeping track of just a packet loss rate and adjusting for error probability for the path's number of hops.

Similarly, the RAP architecture [12] tries to provide real-time communication in large-scale WSNs. By giving higher priority to packets with longer routing distances the architecture's scheduling reduces deadline misses for packets far-away from their destination.

The paper in [13] presents a middleware mechanism that allows users of the network to request different QoS requirements. It also considers that stricter QoS requirements can be achieved by exploiting the redundancy of WSNs.

Various QoS aware routing protocols exist, which try to achieve the necessary quality while still conserving energy. Examples include the protocols published in [14] and [15].

In this paper we implicitly assumed that a sensor network implementing the proposed mechanism would be able to measure the relevant QoS metrics, such as packet latencies. One common way to achieve those capabilities is to have a notion of global time. The work in [16] proposes a way to achieve efficient global clock synchronization for WSNs.

B. Reliable Wireless Communication

The study of reliable wireless communication in the context of WSNs has investigated several issues. One problem that has been identified is that, as the popularity of WSNs and other networks grows, the existing electromagnetic spectrum will get crowded [17].

To some extent the problem is already being observed in existing environments. Experiments referred in the work of [4] identify a packet loss between 3% and 58% for a multi-hop 802.15.4 sensor network, sharing the 2.4 GHz frequency with a WiFi network, depending on the sending rate of the competing WiFi network and the length of the WSN routing path.

The problem can be dealt from different perspectives. One, particularly relevant to this paper, is to allow the networks to cooperatively allocate resources to control QoS among themselves [17]. Another solution, also relevant for reliability, is to increase the redundancy of communication, allowing nodes to operate over various communication channels and frequencies [18], [19] or even providing them with multiple radios [20].

Another issue that has received attention is understanding and modeling the propagation of radio waves. Higher accuracy models have been devised [21], [22], which are relevant for highly tuned QoS-driven architectures and for simulations to validate their results [23]. Correct models are necessary to understand the causes of packet delivery failure [24], the probability of failure [25] and consequences on multi-path routing [26].

Also relevant to dependable adaptation is how to assess the wireless link quality [27], [28], [29] and how to detect the occurrence of radio interferences [30], which is necessary to trigger

adaptation.

VI. Concluding Remarks

We presented a lightweight solution for the dependable characterization of network QoS metrics, based on non-parametric statistics, which allows WSNs and their applications to dependably adapt to changing conditions.

We evaluated the complexity and performance of the proposed solution, concluding it to be lightweight enough for WSNs. We also benchmarked its adaptation effectiveness. The benchmark validated the capability of the proposed technique to successfully adapt in a variety of real-world conditions.

Acknowledgment

The authors would like to thank Teresa Alpuim for her support.

References

[1] Dixit, M., Casimiro, A., Lollini, P., Bondavalli, A., Verissimo, P.: Adaptare: Supporting automatic and dependable adaptation in dynamic environments. ACM Transactions on Autonomous and Adaptive Systems **(to appear)** (2011) also as Technical Report DI/FCUL TR-09-19.

[2] Zhou, G., He, T., Stankovic, J.A., Abdelzaher, T.: Rid: Radio interference detection in wireless sensor networks. In: in INFOCOM. (2005)

[3] Xu, W., Trappe, W., Zhang, Y.: Channel surfing: defending wireless sensor networks from interference. In: Information Processing in Sensor Networks. (2007) 499–508

[4] Musaloiu-elefteri, R., Terzis, A.: Minimising the effect of wifi interference in 802.15.4 wireless sensor networks. International Journal of Sensor Networks **3** (2008) 43–54

[5] Blum, M., Floyd, R.W., Pratt, V.R., Rivest, R.L., Tarjan, R.E.: Time bounds for selection. Technical report, Stanford, CA, USA (1973)

[6] Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. The MIT Press, New York (2001)

[7] Martínez, J.F., Garcí, A.B., Corredor, I., López, L., Hernández, V., Dasilva, A.: Qos in wireless sensor networks: survey and approach. In: Proceedings of the 2007 Euro American conference on Telematics and information systems. EATIS '07, New York, NY, USA, ACM (2007) 20:1–20:8

[8] Akyildiz, I.F., Melodia, T., Chowdhury, K.R.: A survey on wireless multimedia sensor networks. Computer Networks **51** (2007) 921–960

[9] Younis, M., Akkaya, K., Eltoweissy, M., Wadaa, A.: On handling qos traffic in wireless sensor networks. Hawaii International Conference on System Sciences **9** (2004) 90292a

[10] Li, S., Neelisetti, R., Liu, C.: Efficient multi-path protocol for wireless sensor networks. International Journal of Wireless and Mobile Networks (IJWMN) (Jan 2010)

[11] Felemban, E., gun Lee, C., Ekici, E., Boder, R., Vural, S.: Probabilistic qos guarantee in reliability and timeliness domains in wireless sensor networks. In: Proc. of the IEEE Infocom. (2005) 2646–2657

[12] Lu, C., Blum, B., Abdelzaher, T., Stankovic, J., He, T.: Rap: A real-time communication architecture for large-scale wireless sensor networks. In Real-Time Technology and Applications Symposium (2002)

[13] Sharifi, M., Taleghan, M.A., Taherkordi, A.: A middleware layer mechanism for qos support in wireless sensor networks. Mobile Communications and Learning Technologies, Conference on Networking, Conference on Systems, International Conference on **0** (2006) 118

[14] Mahapatra, A., Anand, K., Agrawal, D.: Qos and energy aware routing for real-time traffic in wireless sensor networks. Computer Communications (Jan 2006)

[15] Akkaya, K., Younis, M.: Energy and qos aware routing in wireless sensor networks. Cluster Computing (Jan 2005)

[16] Li, Q., Rus, D.: Global clock synchronization in sensor networks. IEEE Transactions on Computers **55**(2) (2006)

[17] Zhou, G., Stankovic, J.A., Son, S.H.: Crowded spectrum in wireless sensor networks. In: Proceedings of Third Workshop on Embedded Networked Sensors (EmNets. (2006)

[18] Xu, W., Trappe, W., Zhang, Y.: Channel surfing: defending wireless sensor networks from interference. In: Information Processing in Sensor Networks. (2007) 499–508

[19] Kim, Y., Shin, H., Cha, H.: Y-mac: An energy-efficient multi-channel mac protocol for dense wireless sensor networks. In: Information Processing in Sensor Networks. (2008) 53–63

[20] Ansari, J., Zhang, X., Mähönen, P.: Multi-radio medium access control protocol for wireless sensor networks. In: Conference On Embedded Networked Sensor Systems. (2007)

[21] Scott, T., Wu, K., Hoffman, D.: Radio propagation patterns in wireless sensor networks: new experimental results. In: Proceedings of the 2006 international conference on Wireless communications and mobile computing. 857–862

[22] Zhou, G., He, T., Krishnamurthy, S., Stankovic, J.A.: Models and solutions for radio irregularity in wireless sensor networks. ACM Transactions on Sensor Networks **2** (2006) 221–262

[23] Martınez-Sala, A., Molina-Garcıa-Pardo. . . , J.: An accurate radio channel model for wireless sensor networks simulation. Journal of Communications and Networks (Jan 2005)

[24] Srinivasan, K., Dutta, P., Tavakoli, A., Levis, P.: Understanding the causes of packet delivery success and failure in dense wireless sensor networks. Technical report, In Technical report SING-06-00 (Jan 2006)

[25] Cerpa, A., Wong, J.L., Kuang, L., Potkonjak, M., Estrin, D.: Statistical model of lossy links in wireless sensor networks. In: Information Processing in Sensor Networks. (2005) 81–88

[26] Cerpa, A., Wong, J.L., Potkonjak, M., Estrin, D.: Temporal properties of low power wireless links: modeling and implications on multi-hop routing. In: Mobile Ad Hoc Networking and Computing. (2005) 414–425

[27] Baccour, N., Koubía, A., Ben Jamía, M., do Rosário, D., Youssef, H., Alves, M., Becker, L.B.: Radiale: A framework for designing and assessing link quality estimators in wireless sensor networks. Ad Hoc Netw. **9** (September 2011) 1165–1185

[28] Baccour, N., Koubaa, A., Jam Atextcenta, M.B., Youssef, H., Zuniga, M., Alves, M.: A comparative simulation study of link quality estimators in wireless sensor networks. (2009) 1–10

[29] Xu, Y., chien Lee, W.: Exploring spatial correlation for link quality estimation in wireless sensor networks. In: in Proc. IEEE PerCom. (2006) 200–211

[30] Zhou, G., He, T., Stankovic, J.A., Abdelzaher, T.: Rid: Radio interference detection in wireless sensor networks. In: in INFOCOM. (2005)